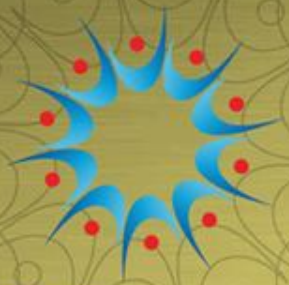


OVER A DECADE
OF EXCELLENCE

Under the hood testing
- Code Reviews -

PALADION

- Harshvardhan Parmar

In the news…

September 2011

- A leading bank's Database hacked (SQLi)

June 2011

- Sony hack exposes consumer passwords (SQLi)

April 2011

- Sony sites offline after Anonymous attack threats (DoS)

March 2011

- Facebook XSS flaw misused for automatic Wall posting (XSS/CSRF)
- MySQL.com hacked (SQLi)

January 2011

- DNS Hack Brings Down Google Bangladesh For Many (DNS Hijacking)

Why?!

- Are these new/unknown attacks?
  - No, all of them are in fact very common attack vectors
- Do Sony, Google and others affected entities not take security seriously?
  - On the contrary, they spend millions on security testing

How people assess application security

- Testing the web app for vulnerabilities via the application interface
- Checking for known issues in the underlying technology used (web servers, DB servers, application framework, etc)
- Checking proper security configurations (encryption, password policies, etc)

Then why are they still vulnerable??

- Simple answer
  - It is not enough
- Why is it not enough?
  - Too many parameters to test manually
  - Automated scanners have limited ability

So what next?

Introducing…

CODE REVIEWS

- Involves reviewing the code for possible vulnerabilities

Code Review – Triggers

- Risk assessment process / security policy dictates a source code review is required
- Regulatory compliance requirement Ex. PCI DSS section 6.3.7
- In response to a security incident

Code Review – Objectives

- To assure appropriate security is built in to high risk applications
- To enable the continual improvement of secure software development practices

Code Review – Benefits

- Consistently proven to be much more exhaustive
- Avoids the security risks and disruption associated with exploitative penetration testing
- Faster, more efficient and lower cost

Reviewing the code

- Context
- Architecture
- Threat Profile

Reviewing the code

- Threat Profile
  - Collection of all possible threats
- Generation: Take inputs from
  - Software Requirements Specification
  - Security Requirements identified in the Design stage
  - Security Standards like OWASP ASVS
- Example: View the account details of another user

Reviewing the code

- Context
- Architecture
- Threat Profile
- Abuse Cases

Reviewing the code

- Abuse Cases
  - Tests/exploits to check whether a given threat can be realized or not
- Well-known cases
  - Ex. SQL Injection, Cross Site Scripting
- Custom cases
  - Ex. For a Funds Transfer Page:

    “changing the source account number to another person’s”

Reviewing the Code

- Context
- Architecture
- Threat Profile
- Abuse Cases
- Scans

Reviewing the Code

- Scans

	Dynamic	Static
Automatic	Web App Scan	Source Code Scan
Manual	Web App Security Test	Code Review

Manual V/s Automated Code Review

Source Code Scan

1. Pattern matching in source code
   - Look for vulnerability pattern
   - Look for coding errors
2. Analysis of Coding Syntax to give coding errors
3. Data path analysis – give an Input to output path mapping.
4. Statistical analysis – number of vulnerabilities per website etc.
5. Unaware of Context - All code scanners are unaware of business logic flaws.

Manual Code Review

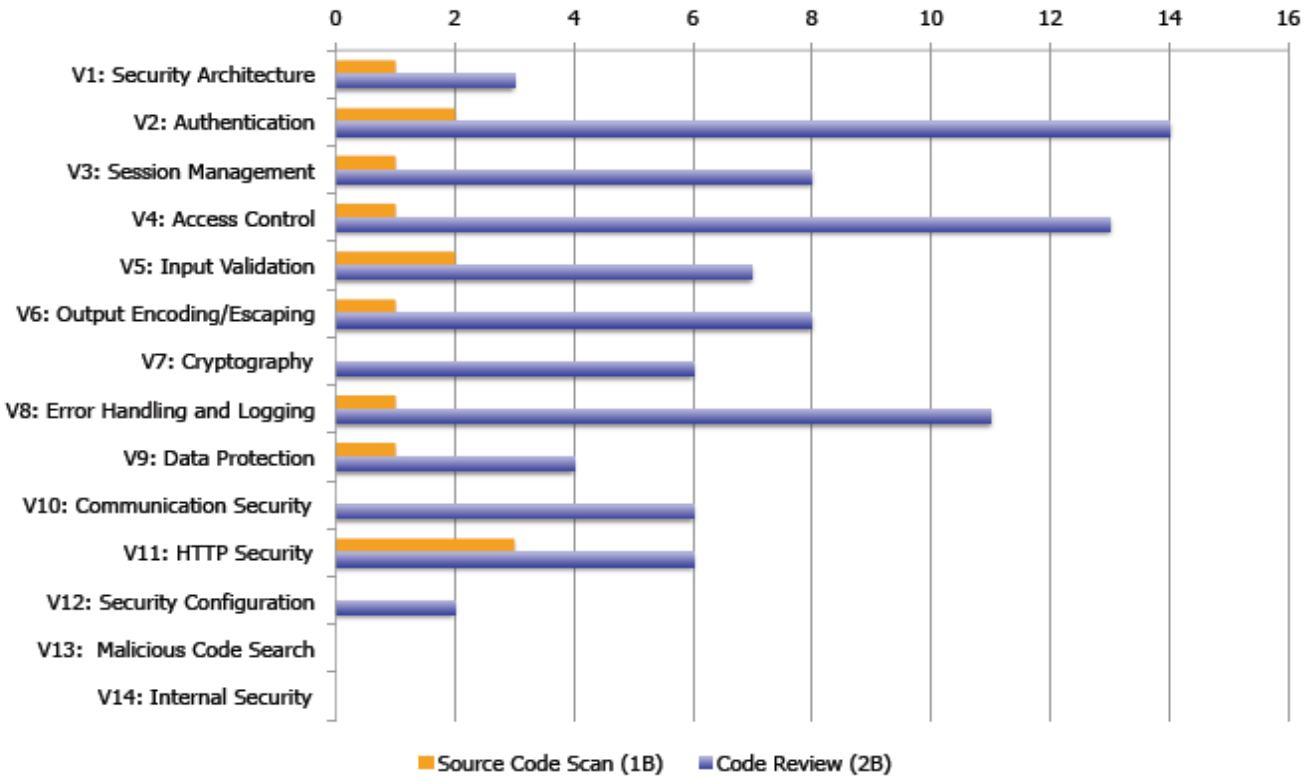
1. Understand the context of the application
   - Prepare a threat profile
   - Define the attack surface
2. Identify key vulnerabilities using simple text matching techniques – example:- grep
3. Understand application configuration flaws
4. Perform logic validation
   - Authentication logic
   - Authorization logic
   - Custom security constraints – approval procedure
   - Design analysis

Manual V/s Automated Code Review

Source Code Scan	Manual Code Review
Benefits 1. Fast 2. Covers a baseline of vulnerabilities 3. Easily repeatable 4. Gives a brief suggestion on fixing vulnerabilities 5. Gives vulnerability statistic analysis	**Benefits** 1. More precise or accurate assessments 2. Threat based analysis ensures that reviewer does not miss any class of vulnerability or section of code. 3. Insight into design and overall quality of the application 4. Precise recommendations on fixing vulnerabilities

Manual V/s Automated Code Review

Quasi-scientific quantitative matrix analysis



Ari Kesaniemi in his presentation to OWASP in November of 2009

What to choose?

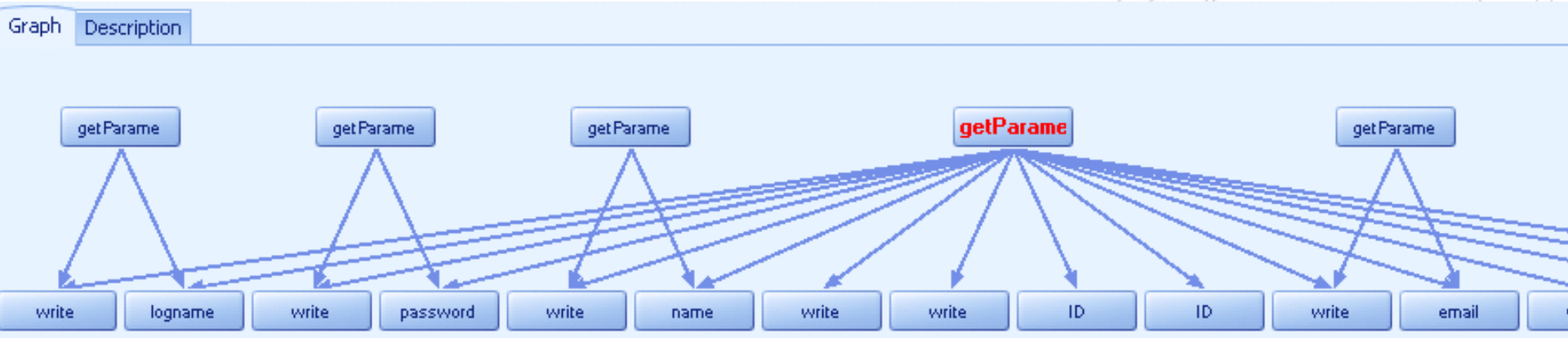
- A hybrid approach of automated as well as manual static scans

Performing Code Reviews

Finding vulnerabilities

- Use a hybrid approach to find vulnerabilities during Code Review
- Use scanners to identify pattern-based vulnerabilities
- Use manual review to identify flaws in business logic

Data Flow – Source to Sink

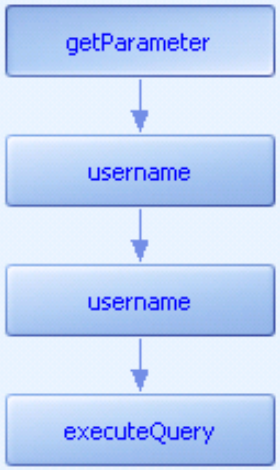


PALADION

Example 1 – SQL Injection

login.jsp | catedelete.jsp | commentnew.jsp | useredit.jsp | catenew.jsp | infodetail.jsp | infoattach.jsp | infoedit.jsp

```
48        boolean isCheckOK;
49        isCheckOK=false;
50        username=request.getParameter("textfield");
51        pwd=request.getParameter("textfield2");
52        if (username==null)
53            username="";
54        if (pwd==null)
55            pwd="";
56        if (username.equals("") || pwd.equals(""))
57          out.println("<font color='#FF0000'>Please Enter Username or Password</font>");
58        else
59        {
60          rst=stmt.executeQuery("select logname,password,usertype from userinfo where log
61
62
63          if (rst!=null && !rst.next())
64            {
65            out.println("<font color='#FF0000'>Invalid Login</font>");
66            }
```



PALADION

Dynamic query

login.jsp | catedelete.jsp | commentnew.jsp | useredit.jsp | catenew.jsp | infodetail.jsp | infoattach.jsp | infoedit.jsp

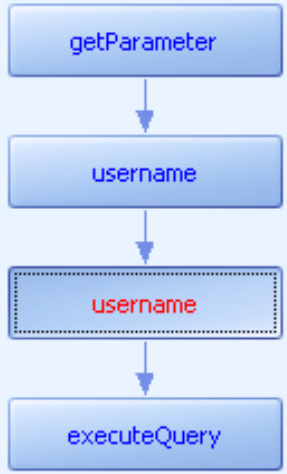
```
ameter("textfield");
r("textfield2");


 || pwd.equals(""))
lor='#FF0000'>Please Enter Username or Password</font>");

("select logname,password,usertype from userinfo where logname='"+ username +"' and pas

.next())

color='#FF0000'>Invalid Login</font>");
```

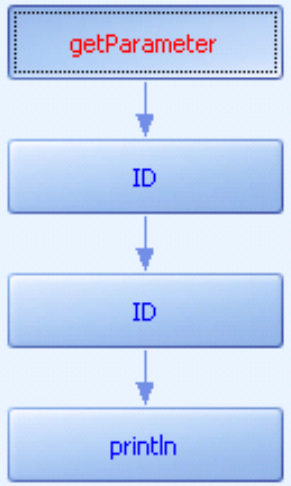


PALADION

Example 2 – File Download

login.jsp | catedelete.jsp | commentnew.jsp | useredit.jsp | catenew.jsp | infodetail.jsp | infoattach.jsp | infoedit.jsp

```
26    ID=request.getParameter("ID") ;
27    if (ID==null) ID="0";
28
29    int pagesize,recount=0;
30      String id1,id2,pageset,direct;
31      boolean isDirOK;
32      isDirOK=true;
33      pagesize=10;
34      id1=request.getParameter("id1");
35      if (id1==null) id1="0";
36      id2=request.getParameter("id2");
37      if (id2==null) id1="0";
38      direct=request.getParameter("direct");
39      if (direct==null) direct="0";
40      pageset=request.getParameter("set");
41      if (pageset==null) pageset="";
42
43      Statement stmt1 = conn.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,ResultSet
44      ResultSet rst1=null;
```

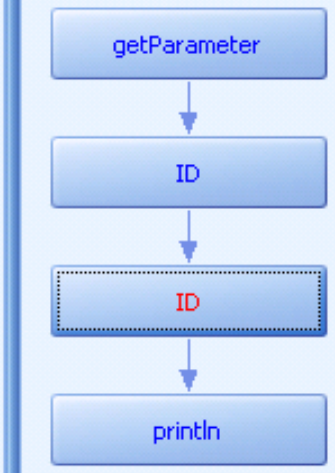


PALADION

No input validation

login.jsp | catedelete.jsp | commentnew.jsp | useredit.jsp | catenew.jsp | infodetail.jsp | infoattach.jsp | infoedit.jsp

```
97      </tr>
98       <tr align="center">
99        <td colspan="2">
100         <p><%
101       String fPath;
102       String fileName=ID +".pdf";
103       fPath=request.getRealPath("/attach/");
104       //out.println(fpath);
105       File fileAtt=new File(fPath,fileName);
106       if (fileAtt.exists()){
107           out.println("<a href='attach\\"+ ID +".pdf'>Download</a>");
108       }
109       else
110       { out.println(" No Table of Contents Available for download"); }
111
112       %> </p></td>
113     </tr>
114     <tr align="center">
115       <td colspan="2">
```

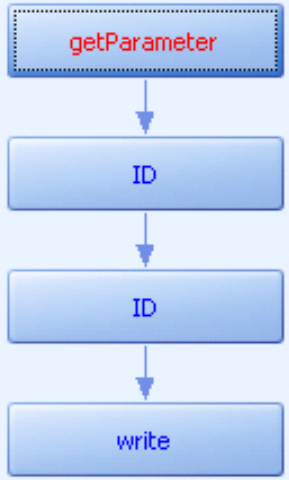


PALADION

Example 3 - XSS

login.jsp | catedelete.jsp | commentnew.jsp | useredit.jsp | catenew.jsp | infodetail.jsp | infoattach.jsp | infoedit.jsp

```
21    ID=request.getParameter("ID");
22    if (ID==null) ID="0";
23    ResultSet rst=null;
24    Statement stmt=conn.createStatement();
25    if (Integer.parseInt(ID)>0 && OPStr.equals("delete"))
26    {
27      stmt.executeUpdate("delete from [info] where ID=" + ID);
28      conn.close();
29      out.println("<div align='center'>Deleting...<a href='infomanager.jsp'>Catalog Mana
30      out.close();
31    }
32    String rstVal="";
33    if (Integer.parseInt(ID)>0 && OPStr.equals("edit"))
34    {
35         rst=stmt.executeQuery("select [name],[author],[keyword],[catename],[content] fr
36        if (rst!=null && rst.next())
37        {
38            title=rst.getString("name");
39            if (title==null) title="";
```

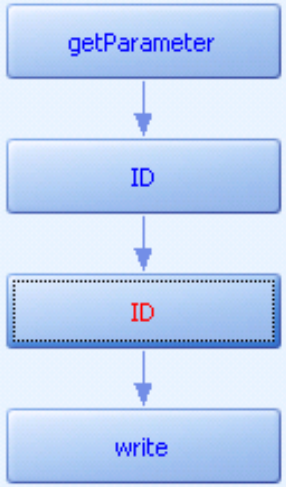


PALADION

Improper data handling

login.jsp | catedelete.jsp | commentnew.jsp | useredit.jsp | catenew.jsp | infodetail.jsp | infoattach.jsp | infoedit.jsp

```
107                rst.close();
108                %>
109              </select></td>
110          <td width="80"> </td>
111      </tr>
112         <tr align="center">
113            <td width="80" height="150">Content</td>
114            <td width="80"><textarea name="textarea" cols="45" rows="10"><%=content%></tex
115          <td width="80"> </td>
116      </tr>
117    </table>
118
119       <input name="buttonl" type="submit" id="buttonl" value="Save">
120        
121       <a href="infoedit.jsp?ID=<%=ID%>&OP=delete">Delete Record</a> <a href="infoattach.
122      </form>
123      <p> </p>
124      <%
125      boolean isUpdateOK=false;
```



PALADION

Manual Review

- Look for business logic flaws
- Example
  - If the application has a feature which allows you to transfer funds, check whether it validates the account balance before performing the funds transfer

Implementing mitigations

- Short term & Long term
- Refer:
  - ESAPI [OWASP]
  - Microsoft Enterprise Library
  - Core security patterns

Inference

- Code Reviews help in identifying vulnerabilities that would be missed during a web application assessment
- For a new application, a Code Review will help in ensuring the development of a secure application
- For existing applications, a Code Review helps in comprehensively identifying vulnerabilities at the code level

Some Code Scanners

- Non-Commercial download
  - LAPSE - http://suif.stanford.edu/~livshits/work/lapse/
  - FxCOP - http://www.gotdotnet.com/Team/FxCop/
  - RATS - http://www.fortify.com/security-resources/rats.jsp
- Commercial Scanners
  - CheckMarx - http://www.checkmarx.com/
  - Fortify 360 - http://www.fortify.com/products/detect/
  - Klocwork - www.klocwork.com

Thank You!

Harshvardhan Parmar – Project Manager, Paladion

harshvardhan.p@paladion.net

